# Backuppy Documentation

***Release master***

**Bart Feenstra**

**Mar 18, 2018**

# Contents:

CHAPTER 1

# Indices and tables

- genindex
- modindex
- search

CHAPTER 2

# Backuppy

## 2.1 About

Backuppy backs up and restores your data using rsync, allowing different routes to the same, or different destinations.

The following instructions can be executed in any system Python environment, but you may want to use a virtual environment. Alternatively, some actions can be performed using tox as well, which produces its own virtual environments in `.tox/py**`.

## 2.2 License

Backuppy is released under the MIT license.

## 2.3 Documentation

Extended API documentation is available at Read the Docs.

## 2.4 Usage

### 2.4.1 Requirements

- Python 2.7+

### 2.4.2 Installation

In any Python environment, run `pip install backuppy`.

### 2.4.3 Command line

```
$ backuppy --help
usage: backuppy [-h] {backup,restore,init} ...

Backuppy backs up and restores your data using rsync.

positional arguments:
  {backup,restore,init}
    backup              Starts a back-up.
    restore             Restores a back-up.
    init                Initializes a new back-up configuration.

optional arguments:
  -h, --help            show this help message and exit
```

### 2.4.4 Configuration

Configuration is written in YAML or JSON, and can be stored anywhere as `*.yml`, `*.yaml`, or `*.json` files. An example of the smallest possible configuration file:

```
source:
  type: path
  configuration:
    path: ./source
target:
  type: path
  configuration:
    path: ./target
```

To create a new back-up configuration without writing code, run `backuppy init` and follow the interactive wizard.

To tweak Backuppy's output:

```
# An optional human-readable name for this back-up. It will default to the␣
↪configuration file name.
name: Test
# Whether or not to generate verbose output. Defaults to `false`.
verbose: true
# Python logging configuration. This is specific to the Python version you are using.␣
↪Defaults to `null` for no logging.
# See https://docs.python.org/3.6/library/logging.config.html#logging-config-
↪dictschema.
logging:
  version: 1
  # ...additional logger and handler configuration.
```

Backuppy supports plugins for back-up source and target locations, as well as notifications.

To configure a local path-based target:

```
type: path
configuration:
  # The local path to the target directory. If the path is relative, it will be␣
↪resolved against the location of
  # the configuration file.
  path: ./target
```

To configure a remote target over SSH:

```
type: ssh
configuration:
  # The host to connect to.
  host: example.com
  # The SSH port to use. Defaults to 22.
  port: 22
  # The name of the user on the remote system to log in as.
  user: bart
  # The absolute path to the target directory on the remote.
  path: /home/bart/target
```

The SSH key must have been accepted already, and the host must support Bash.

To specify multiple routes to the same target, such as one over a local network mount, and a fallback over SSH:

```
target:
  type: first_available
  configuration:
    targets:
      - type: path
        configuration:
          path: ./target
      - type: ssh
        configuration:
          host: example.com
          user: bart
          path: /home/bart/target
```

To configure user-facing notifications:

```
# An optional list of zero or more notification methods. Message types are:
# - "state": unimportant, mass-generated, or debugging output which may be ignored.
# - "inform": informative messages, such as those marking the start of an action.
# - "confirm": confirmation messages, such as those marking the successful completion␣
↪of an action.
# - "alert": important messages that warrant someone's attention, such as in case of␣
↪errors.
notifications: []
```

To display notifications to stdout and stderr (terminal output):

```
notifications:
  - type: stdio
```

To display notifications using `notify-send`:

```
notifications:
  - type: notify-send
```

To process notifications through custom CLI commands:

```
notifications:
  - type: command
    # Commands are specified as CLI arguments. `fallback` is required if any of the␣
↪others are missing.
    configuration:
      state:
        - echo
        - "{message}"
      inform:
        - echo
        - "{message}"
      confirm:
        - echo
        - "{message}"
      alert:
        - echo
        - "{message}"
      fallback:
        - echo
        - "{message}"
```

To append notifications to files:

```
notifications:
  - type: file
    # Paths must be absolute. `fallback` is required if any of the others are missing.
    configuration:
      state:
        - /var/log/backuppy
      inform:
        - /var/log/backuppy
      confirm:
        - /var/log/backuppy
      alert:
        - /var/log/backuppy
      fallback:
        - /var/log/backuppy
```

## 2.5 Development

### 2.5.1 Requirements

- The generic requirements documented earlier.
- Bash (you're all good if `which bash` outputs a path in your terminal)

### 2.5.2 Installation

Run `git clone https://github.com/bartfeenstra/backuppy.git`.

If you wish to contribute code changes, you may want to fork this project first, and clone your own forked repository instead.

### 2.5.3 Building

In any Python environment, run `./bin/build-dev`.

With tox, run `tox --develop --notest`.

### 2.5.4 Testing

In any Python environment, run `./bin/test`.

With tox, run `tox --develop`

### 2.5.5 Viewing documentation

In any Python environment, run `./bin/build-docs`, and open `./docs-build/index.html`.

### 2.5.6 Fixing problems automatically

In any Python environment, run `./bin/fix`.